

데이터베이스

관계형 데이터베이스

03. SELECT로 특정 데이터를 추출하기

first
coding

01 WHERE 조건과 비교 연산자

- 원하는 행만 얻으려면 다음과 같이 행을 제한하는 조건을 SELECT 문에 WHERE 절을 추가 해야 합니다.

형식

```
SELECT * [column1, column2, .. ,columnn]  
FROM table_name  
WHERE 조건절;
```

- 조건절은 다음의 세부분으로 구성이 됩니다.

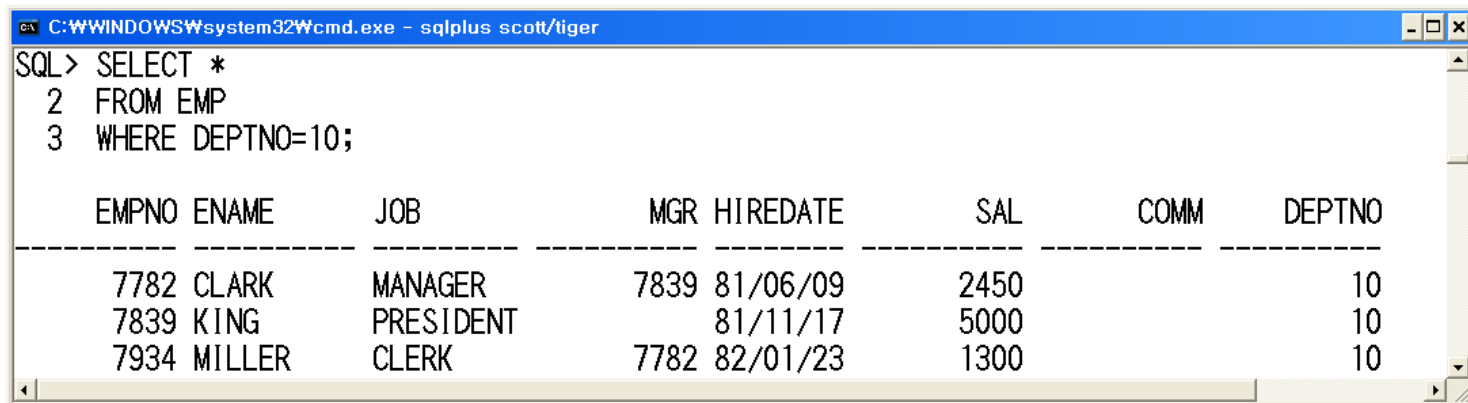
조건절의 구성

WHERE SAL >= 3000;
 ① 컬럼 ② 연산자 ③ 비교대상값

연산자 종류	설 명
=	같은 조건을 검색
!= , <>	같지 않은 조건을 검색
>	큰 조건을 검색
>=	크거나 같은 조건을 검색
<	작은 조건을 검색
<=	작거나 같은 조건을 검색
BETWEEN a AND b	A 와 B사이에 있는 범위 값을 모두 검색
IN(a,b,c)	A 이거나 B 이거나 C 인 조건을 검색
Like	특정 패턴을 가지고 있는 조건을 검색
Is Null / Is Not Null	Null 값을 검색 / Null 이 아닌 값을 검색
A AND B	A 조건과 B 조건을 모두 만족하는 값만 검색
A OR B	A 조건이나 B 조건 중 한가지라도 만족하는 값을 검색
NOT A	A 가 아닌 모든 조건을 검색

예

```
SELECT *  
FROM EMP  
WHERE DEPTNO=10;
```



EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7782	CLARK	MANAGER	7839	81/06/09	2450		10
7839	KING	PRESIDENT		81/11/17	5000		10
7934	MILLER	CLERK	7782	82/01/23	1300		10

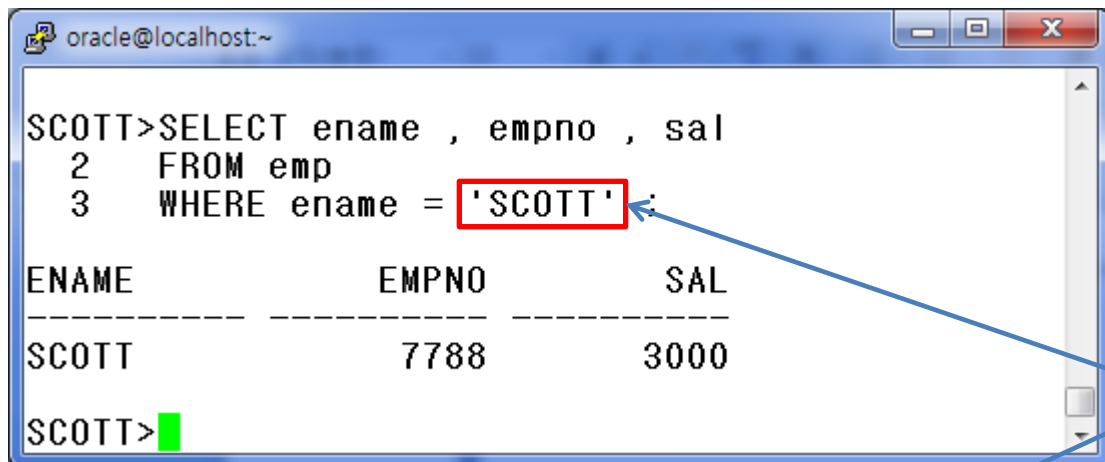
- 이번에는 급여(숫자)가 아닌 사원 이름 같은 문자 데이터를 조회해 봅시다.
- 다음은 이름이 FORD인 사원의 사원번호(EMPNO)와 사원이름(ENAME)과 급여(SAL)을 출력하는 예제

예

```
SELECT EMPNO, ENAME, SAL  
FROM EMP  
WHERE ENAME='FORD';
```

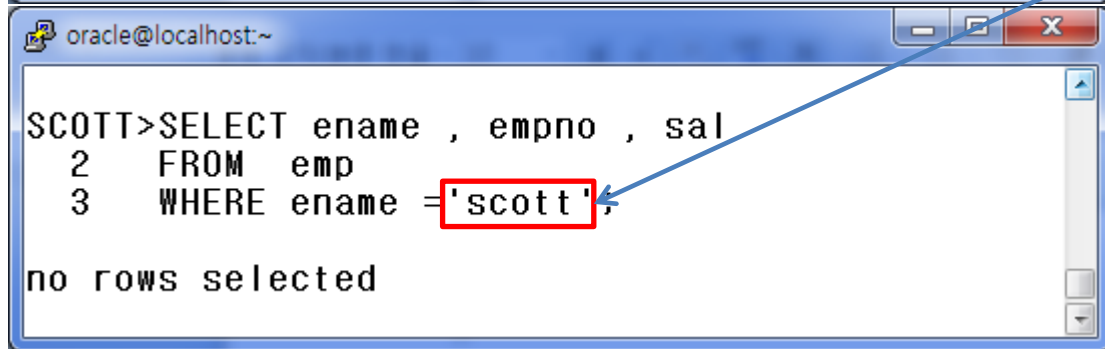
- SQL에서 문자열이나 날짜는 반드시 단일 따옴표(single quotation) 안에 표시해야 합니다.
- SQL문에 사용되는 키워드인 SELECT 나 FROM 이나 WHERE 등은 대소문자를 구별하지 않지만 테이블 내에 저장된 데이터 값은 대소문자를 구분하기에 WHERE ENAME='ford'와 같이 기술하면 사원이름이 FORD 인 사원을 찾을 수 없습니다.

- 문자열 조회할 때 주의 사항



```
oracle@localhost:~  
SCOTT>SELECT ename , empno , sal  
2 FROM emp  
3 WHERE ename = 'SCOTT';  
  
ENAME          EMPNO          SAL  
-----  
SCOTT           7788           3000  
  
SCOTT>
```

WHERE 절의 문자는
대소문자 구분 한다!
홀 따옴표로 묶는다!



```
oracle@localhost:~  
SCOTT>SELECT ename , empno , sal  
2 FROM emp  
3 WHERE ename = 'scott';  
  
no rows selected
```

- 날짜 조회할 때 주의 사항

```
oracle@localhost:~  
SCOTT>SELECT name , pay , hiredate  
2 FROM professor  
3 WHERE hiredate = 23-MAY-01;  
WHERE hiredate = 23-MAY-01  
*  
ERROR at line 3:  
ORA-00904: "MAY": invalid identifier
```

- 홀 따옴표로 묶는다
- 대소문자 구분 없다.

```
oracle@localhost:~  
SCOTT>SELECT name , pay , hiredate  
2 FROM professor  
3 WHERE hiredate = '23-MAY-01';  
  
NAME                PAY HIREDATE  
-----  
허은                290 23-MAY-01
```

```
oracle@localhost:~  
SCOTT>SELECT name , pay , hiredate  
2 FROM professor  
3 WHERE hiredate = '23-may-01';  
  
NAME                PAY HIREDATE  
-----  
허은                290 23-MAY-01
```

- 오라클에서 사용 가능한 논리 연산자 AND나 OR나 NOT가 있습니다.

연산자	의미
AND	두 가지 조건을 모두 만족해야만 검색할 수 있다. SELECT * FROM emp WHERE deptno=10 AND job='MANAGER';
OR	두 가지 조건 중에서 한 가지만 만족하더라도 검색할 수 있다. SELECT * FROM emp WHERE deptno=10 OR job='MANAGER';
NOT	조건에 만족하지 못하는 것만 검색한다. SELECT * FROM emp WHERE NOT deptno=10;

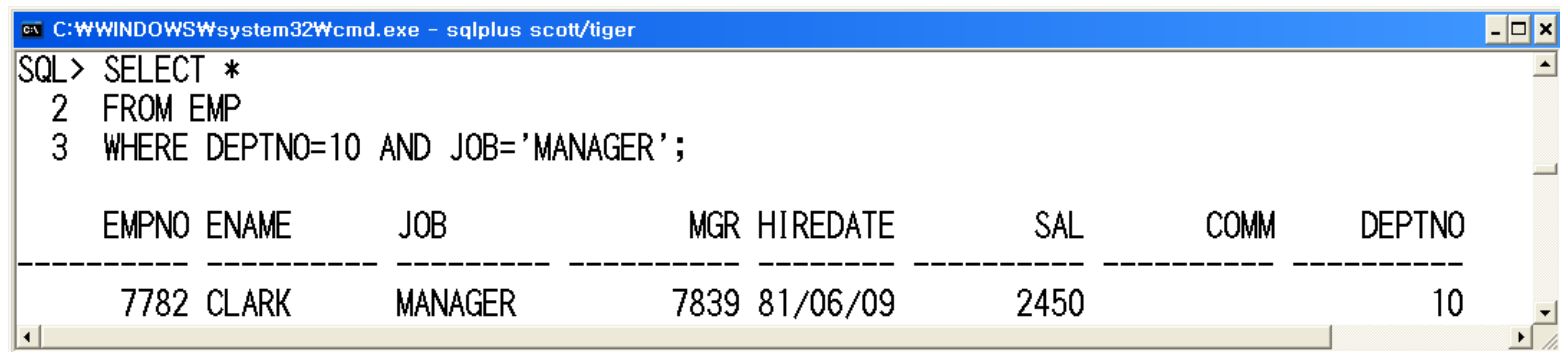
- 10번 부서 소속인 직원들 중에서 직급이 MANAGER인 사람을 검색하여 직원명, 부서번호, 직급을 출력하려고 한다면 두 가지 조건을 제시해야 합니다.

[조건1] 10번 부서 소속인 직원 : DEPTNO=10

[조건2] 직급이 MANAGER인 직원 : JOB='MANAGER'

예

```
SELECT *  
FROM EMP  
WHERE DEPTNO=10 AND JOB='MANAGER';
```



```
C:\WINDOWS\system32\cmd.exe - sqlplus scott/tiger  
SQL> SELECT *  
2 FROM EMP  
3 WHERE DEPTNO=10 AND JOB='MANAGER';  
  
EMPNO ENAME      JOB      MGR HIREDATE      SAL      COMM      DEPTNO  
-----  
7782 CLARK      MANAGER  7839 81/06/09      2450        
10
```

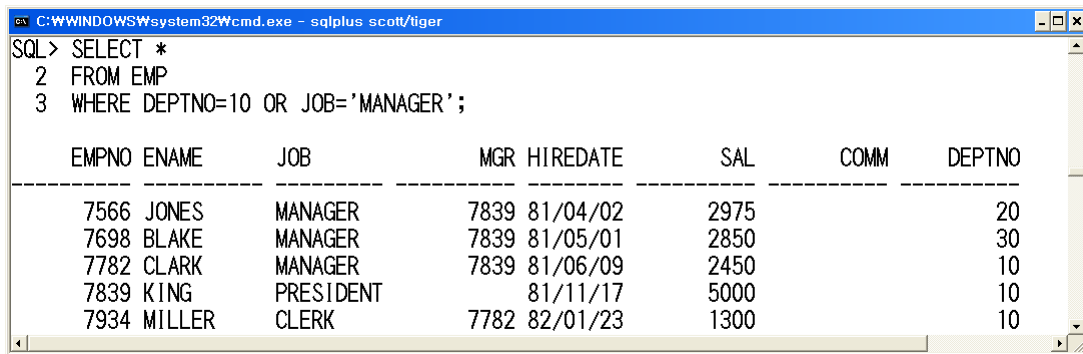
- 10번 부서에 소속된 사원이거나 직급이 MANAGER인 사람을 검색하여 사원명, 부서번호, 직급을 출력합니다.

[조건1] 10번 부서 소속인 사원 : DEPTNO=10

[조건2] 직급이 MANAGER인 사원 : JOB='MANAGER'

예

```
SELECT *  
FROM EMP  
WHERE DEPTNO=10 OR JOB='MANAGER';
```



```
SQL> SELECT *  
2 FROM EMP  
3 WHERE DEPTNO=10 OR JOB='MANAGER';
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7566	JONES	MANAGER	7839	81/04/02	2975		20
7698	BLAKE	MANAGER	7839	81/05/01	2850		30
7782	CLARK	MANAGER	7839	81/06/09	2450		10
7839	KING	PRESIDENT		81/11/17	5000		10
7934	MILLER	CLERK	7782	82/01/23	1300		10

- 이 조건 앞에 NOT을 붙이면 부서번호가 10번이 아닌 직원들에 대해서만 검색하게 됩니다.
- 다음은 부서번호가 10번이 아닌 직원의 직원이름, 부서번호, 직급을 출력해 봅시다.

예

```
SELECT *  
FROM EMP  
WHERE NOT DEPTNO=10;
```

예

```
SELECT *  
FROM EMP  
WHERE DEPTNO <> 10;
```

- 2000에서 3000 사이의 급여를 받는 사원과 같이 특정 범위 내에 속하는 데이터인지를 알아보기 위해서 비교연산자와 논리 연산자를 결합하여 표현할 수 있습니다.

예

```
SELECT *
FROM EMP
WHERE SAL >= 2000 AND SAL <= 3000;
```

C:\WINDOWS\system32\cmd.exe - sqlplus scott/tiger

```
SQL> SELECT *
2 FROM EMP
3 WHERE COMM=300 OR COMM=500 OR COMM=1400;
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30

- 오라클에서는 특정 범위의 값을 조회하기 위해서는 BETWEEN AND 연산자를 사용할 수 있습니다.

형식

column_name **BETWEEN A AND B**

- 다음은 2000에서 3000 사이의 급여를 받는 사원을 조회하기 위해서 BETWEEN AND 연산자를 사용한 예입니다.

예

```
SELECT *  
FROM EMP  
WHERE SAL BETWEEN 2000 AND 3000;
```

- BETWEEN AND 연산자는 숫자형뿐만 아니라 문자형, 날짜형에도 사용할 수 있습니다.
- 주의할 점은 비교 대상이 되는 값을 단일 따옴표로 둘러싸야 한다는 점입니다.
- 1987년에 입사한 사원을 출력해 봅시다.

예

```
SELECT *
FROM EMP
WHERE HIREDATE BETWEEN '1987/01/01' AND '1987/12/31';
```

```
C:\WINDOWS\system32\cmd.exe - sqlplus scott/tiger
SQL> SELECT *
2 FROM EMP
3 WHERE HIREDATE BETWEEN '1987/01/01' AND '1987/12/31';
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7788	SCOTT	ANALYST	7566	87/04/19	3000		20
7876	ADAMS	CLERK	7788	87/05/23	1100		20

04 IN 연산자

- 동일한 필드가 여러 개의 값 중에 하나인 경우인지를 살펴보기 위해서 비교 연산자와 논리 연산자 OR를 사용하여 복잡하게 쿼리문을 작성하지 않고 IN 연산자를 사용하여 훨씬 간단하게 표현할 수 있습니다.

형식

column_name IN(*A*, ***B***, *C*)

- 특정 필드의 값이 A이거나 B이거나 C 중에 어느 하나만 만족하더라도 출력하도록 하는 표현을 IN 연산자를 사용하여 할 수 있습니다.
- 커미션이 300 이거나 500 이거나 1400 인 사원을 검색하기 위해서 IN 연산자를 사용해 봅시다.

예

```
SELECT *  
FROM EMP  
WHERE COMM IN(300, 500, 1400);
```

05 LIKE 연산자와 와일드카드

- LIKE 연산자는 검색하고자 하는 값을 정확히 모를 경우에도 검색 가능하도록 하기 위해서 와일드카드와 함께 사용하여 원하는 내용을 검색하도록 합니다. 다음은 LIKE 연산자의 형식입니다.

형식	<i>column_name</i> LIKE <i>pattern</i>
----	--

- LIKE 다음에는 pattern을 기술해야 하는데 pattern에 다음과 같이 두 가지 와일드카드가 사용됩니다.

와일드카드	의미
%	문자가 없거나, 하나 이상의 문자가 어떤 값이 와도 상관없다.
_	하나의 문자가 어떤 값이 와도 상관없다.

- 찾으려는 이름이 F로 시작 하는 것은 알지만 그 뒤의 문자는 모를 경우 `ename = 'F'`로 검색하게 되면 될까요?
- `ename = 'F'` 표현은 이름이 정확히 F인 사람만을 검색하겠다는 의미이기에 이름이 'F'로 시작하는 사람을 검색하지 못합니다.
- 검색하고자 하는 값을 정확히 모를 경우 즉, 특정 문자 포함되지만 하고 그 이전이나 이후에 어떤 문자가 몇 개가 오든지 상관없다는 의미를 표현하기 위해서는 LIKE 연산자와 함께 %를 사용해야 합니다.

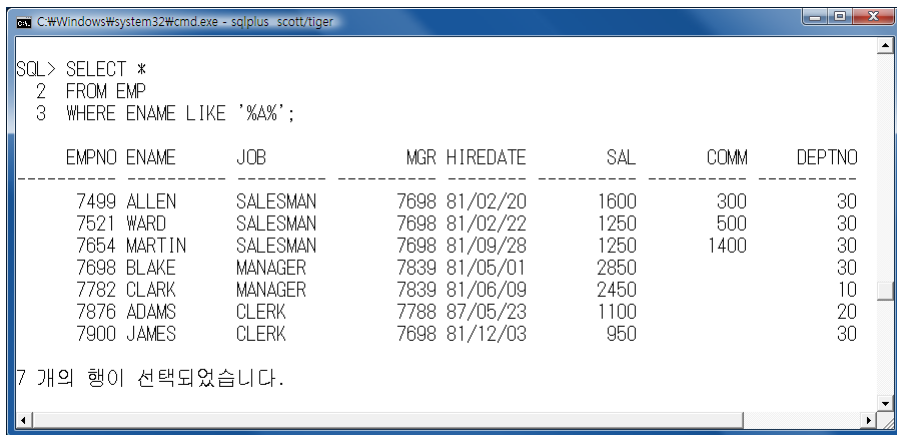
예

```
SELECT *  
FROM EMP  
WHERE ENAME LIKE 'F%';
```

- 이번에는 이름 중 A를 포함하는 사원을 검색해봅시다.
- 문자 A 앞뒤에 %를 기술하면 문자열 중간에 A 문자만 있으면 앞뒤에 어떤 문자열이 몇 개가 오는 상관 없이 찾습니다.

예

```
SELECT *  
FROM EMP  
WHERE ENAME LIKE '%A%';
```



The screenshot shows a SQL*Plus window with the following content:

```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger  
SQL> SELECT *  
2 FROM EMP  
3 WHERE ENAME LIKE '%A%';
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7521	WARD	SALESMAN	7698	81/02/22	1250	500	30
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30
7698	BLAKE	MANAGER	7839	81/05/01	2850		30
7782	CLARK	MANAGER	7839	81/06/09	2450		10
7876	ADAMS	CLERK	7788	87/05/23	1100		20
7900	JAMES	CLERK	7698	81/12/03	950		30

7 개의 행이 선택되었습니다.

- 이번에는 문자열의 앞에 어떤 문자열이 몇 개가 오는 상관없이 N으로 끝나는 데이터를 찾아봅시다.

예

```
SELECT *
FROM EMP
WHERE ENAME LIKE '%N';
```

The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - sqlplus scott/tiger". The user has entered the following SQL query:

```
SQL> SELECT *
2 FROM EMP
3 WHERE ENAME LIKE '%N';
```

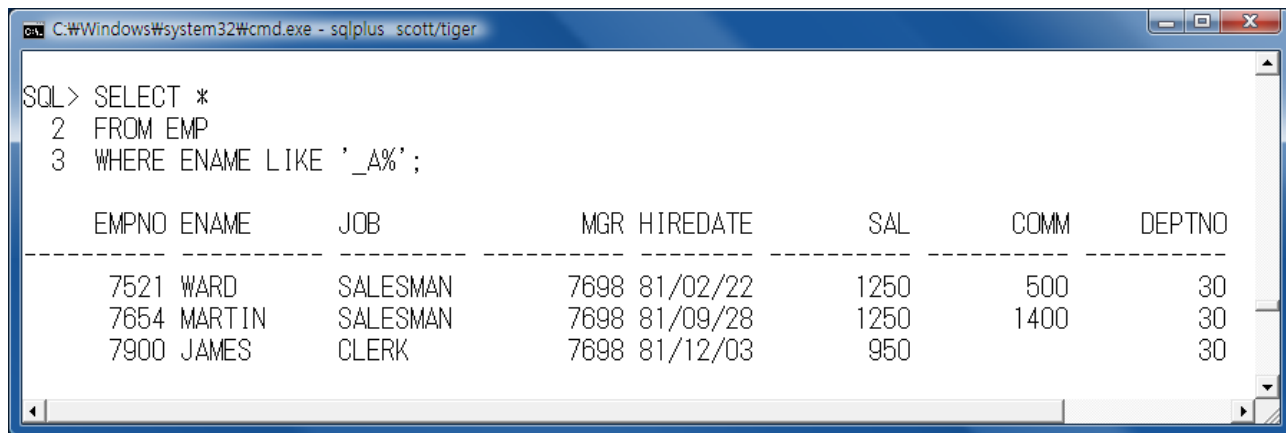
The query results are displayed in a table format with the following columns: EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, and DEPTNO. The results show two employees: ALLEN and MARTIN, both of whom are SALESMAN and work in department 30.

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7499	ALLEN	SALESMAN	7698	81/02/20	1600	300	30
7654	MARTIN	SALESMAN	7698	81/09/28	1250	1400	30

- _ 역시 %와 마찬가지로 어떤 문자가 오는 상관없다는 의미로 사용되는 와일드카드입니다.
- 차이점은 %는 몇 개의 문자가 오는 상관없지만 _ 는 단 한 문자에 대해서만 와일드카드 역할을 합니다.
- 다음은 이름의 두 번째 글자가 A인 사원을 찾는 예제입니다.

예

```
SELECT *  
FROM EMP  
WHERE ENAME LIKE '_A%';
```

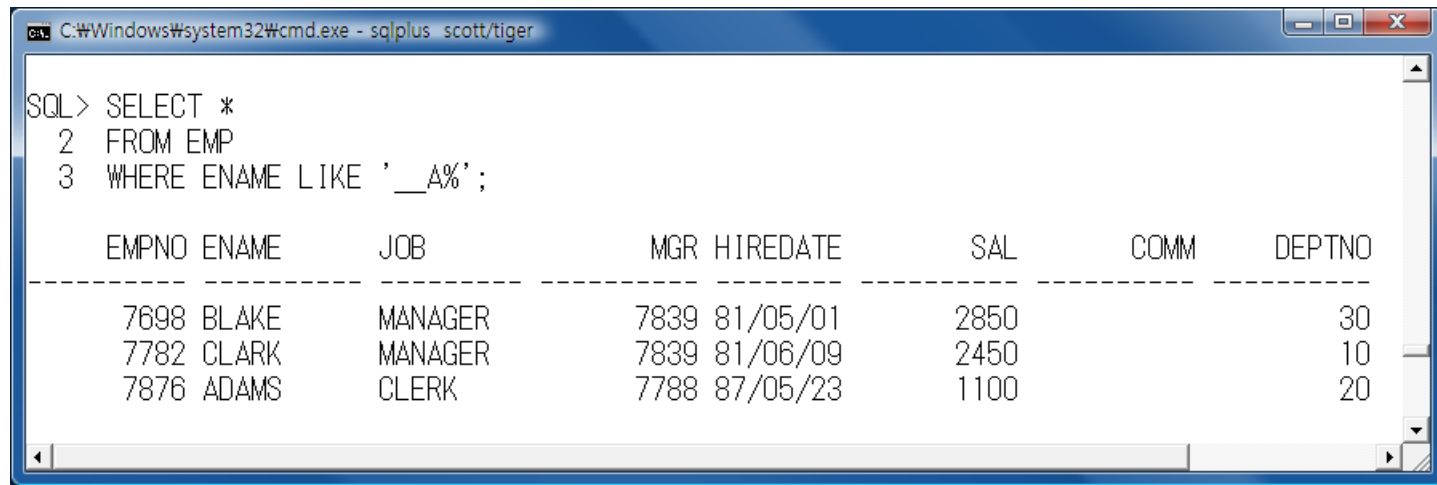


```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger  
  
SQL> SELECT *  
2 FROM EMP  
3 WHERE ENAME LIKE '_A%';  
  
EMPNO ENAME      JOB            MGR HIREDATE          SAL        COMM      DEPTNO  
-----  
7521  WARD          SALESMAN       7698 81/02/22         1250         500         30  
7654  MARTIN        SALESMAN       7698 81/09/28         1250        1400         30  
7900  JAMES         CLERK          7698 81/12/03          950           0         30
```

- 세 번째 글자가 A인 자료를 검색하려면 __A%처럼 기술해야 합니다.

예

```
SELECT *  
FROM EMP  
WHERE ENAME LIKE '__A%';
```



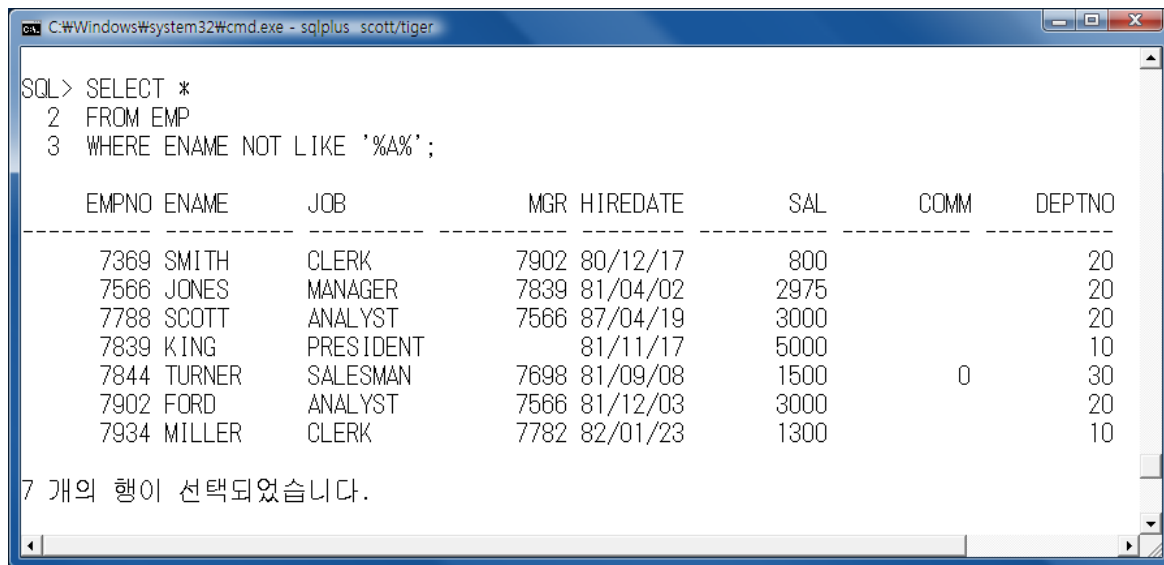
SQL> SELECT *
2 FROM EMP
3 WHERE ENAME LIKE '__A%';

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7698	BLAKE	MANAGER	7839	81/05/01	2850		30
7782	CLARK	MANAGER	7839	81/06/09	2450		10
7876	ADAMS	CLERK	7788	87/05/23	1100		20

- 이름에 A를 포함하지 않은 사람만을 검색하려고 할 경우에 NOT LIKE 연산자를 사용합니다.

예

```
SELECT *  
FROM EMP  
WHERE ENAME NOT LIKE '%A%';
```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - sqlplus scott/tiger". The user has entered the following SQL command:

```
SQL> SELECT *  
2 FROM EMP  
3 WHERE ENAME NOT LIKE '%A%';
```

The output displays a table with 7 rows of employee data. The columns are EMPNO, ENAME, JOB, MGR, HIREDATE, SAL, COMM, and DEPTNO. The data is as follows:

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	80/12/17	800		20
7566	JONES	MANAGER	7839	81/04/02	2975		20
7788	SCOTT	ANALYST	7566	87/04/19	3000		20
7839	KING	PRESIDENT		81/11/17	5000		10
7844	TURNER	SALESMAN	7698	81/09/08	1500	0	30
7902	FORD	ANALYST	7566	81/12/03	3000		20
7934	MILLER	CLERK	7782	82/01/23	1300		10

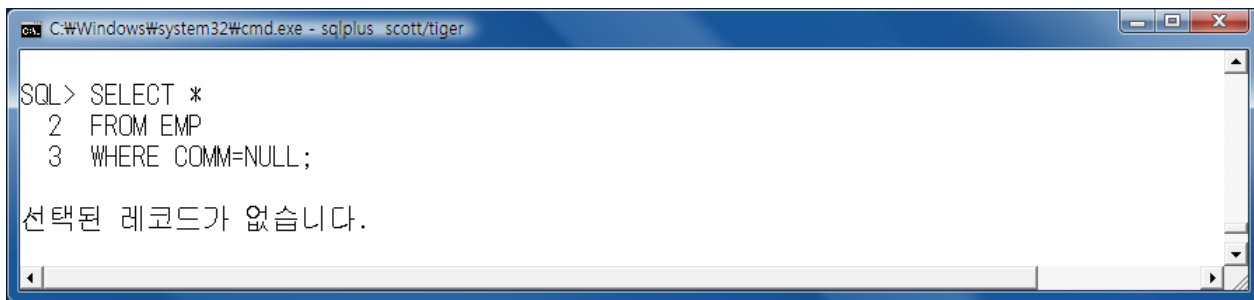
At the bottom of the window, it says "7 개의 행이 선택되었습니다." (7 rows selected).

- 사원 테이블의 커미션 컬럼에 널이 저장되어 있으므로 = 연산자로 커미션을 받지 않는 사원에 대한 검색해 봅시다.

예

```
SELECT *  
FROM EMP  
WHERE COMM=NULL;
```

- NULL 값을 가진 데이터와 비교 연산한 결과는 다음과 같습니다. 왜냐하면 NULL이 저장되어 있는 경우에는 = 연산자로 판단할 수 없기 때문입니다.



```
C:\Windows\system32\cmd.exe - sqlplus scott/tiger  
  
SQL> SELECT *  
2 FROM EMP  
3 WHERE COMM=NULL;  
  
선택된 레코드가 없습니다.
```

- 어떤 컬럼을 NULL 즉, 모르는 값과 같다(=)라는 것은 의미상으로 말이 되지 않기 때문에 = 대신 IS NULL 연산자를 사용해야 합니다.
- IS NULL 연산자 역시 조건절에 사용되면 대상 컬럼과 연산자, 비교할 값 세부분으로 구성되어야 합니다.

형식	대상컬럼 IS (연산자) NULL(비교값)
----	-------------------------

- 이번에는 IS NULL 연산자를 사용하여 커미션을 받지 않는 사원을 검색해 봅시다.

예	SELECT * FROM EMP WHERE COMM IS NULL;
---	---

- 이번에는 커미션을 받는 사원에 대해 검색해 봅시다. 문장대로 해석하면 커미션(COMM) 칼럼이 NULL 아닌 자료만 추출하면 되므로 IS NOT NULL 연산자를 사용하면 됩니다.

형식	<pre>SELECT * FROM EMP WHERE COMM IS NOT NULL;</pre>
----	--

07 정렬을 위한 ORDER BY 절

- 정렬이란 크기 순서대로 나열하는 것을 의미합니다.
 - 오름차순(ascending) 정렬 방식 : 작은 것이 위에 출력되고 아래로 갈수록 큰 값이 출력
 - 내림차순(descending) 정렬 방식 : 큰 값이 위에 출력되고 아래로 갈수록 작은 값이 출력
- 로우를 정렬하기 위해서는 SELECT 문에 ORDER BY 절을 추가하고 어떤 컬럼을 기준으로 어떤 정렬을 할 것인지를 결정해야 합니다.

	ASC(오름차순)	DESC(내림차순)
숫자	작은 값부터 정렬	큰 값부터 정렬
문자	사전 순서로 정렬	사전 반대 순서로 정렬
날짜	빠른 날짜 순서로 정렬	늦은 날짜 순서로 정렬
NULL	가장 마지막에 나온다.	가장 먼저 나온다.

07 1. 오름차순 정렬을 위한 ASC

- 오름차순 정렬은 작은 값부터 큰 값으로 정렬하는 것을 의미합니다.(예:1~9, 'A'~'Z')
- 이를 위해서는 ASC를 칼럼 다음에 기술해야 하는데 만일 생략하게 되면 디폴트로 ASC로 지정되어 있기 때문에 오름차순으로 출력됩니다.
- 다음은 급여 컬럼을 기준으로 오름차순으로 정렬한 예입니다.

예

```
SELECT *  
FROM EMP  
ORDER BY SAL ASC;
```

- 정렬방식을 지정하지 않은 경우에는 디폴트로 오름차순으로 정렬합니다.

예

```
SELECT *  
FROM EMP  
ORDER BY SAL;
```

- 내림차순 정렬은 큰 값부터 작은 값으로 정렬을 하는 것이다.(예:9~1, Z~A)
- 이번에는 급여를 많이 받는 사람부터 적게 받는 사람 순으로 순차적으로 출력해 봅시다.

예

```
SELECT *  
FROM EMP  
ORDER BY SAL DESC;
```

- 큰 값이 위에 출력되고 아래로 갈수록 작은 값이 출력되도록 하려면 내림차순(descending) 으로 정렬해야 하기 때문에 칼럼 다음에 DESC를 기술해야 합니다.

- 크기에 대한 비교는 수치 데이터 뿐만 아니라 문자 데이터나 날짜 데이터에 대해서도 가능합니다.
- 문자 데이터의 경우 아스키 코드 값으로 저장되므로 아스키 코드 값을 기준으로 정렬됩니다.
- 오름차순인 경우에는 A, B, ... Z 순으로 출력되고 내림차순인 경우에는 Z, Y, ... A 순으로 출력됩니다.
- 다음은 사원의 이름을 알파벳 순(오름차순)으로 출력하는 예제입니다.

예

```
SELECT *  
FROM EMP  
ORDER BY ENAME;
```

- 날짜의 경우에도 오름차순 혹은 내림차순으로 출력할 수 있습니다.
- 오름차순으로 지정하면 가장 오래된 과거의 시점이 가장 위에 출력되고 아래로 갈수록 최근 시점이 출력됩니다.
- 내림차순인 경우에는 최근 시점부터 출력합니다.
- 다음은 가장 최근에 입사한 사람부터 출력하는 예제입니다.

예

```
SELECT *  
FROM EMP  
ORDER BY HIREDATE DESC;
```

07 5. 정렬 방식에 여러 가지 조건 제시

- 급여를 많이 받는 사람부터 적게 받는 사람 순으로 순차적으로 출력하는 결과 화면을 살펴보면 동일한 급여를 받는 사람이 존재합니다.
- 급여가 같은 사람이 존재할 경우 이름의 철자가 빠른 사람부터 출력되도록 하려면 정렬 방식을 여러 가지로 지정해야 합니다.

예

```
SELECT *  
FROM EMP  
ORDER BY SAL DESC, ENAME ASC;
```